Headquarters U.S. Air Force

Integrity - Service - Excellence

C4ISR ARCHITECTURES AND A PROCESS FOR THEIR DESIGN

Alexander H. Levis
AF/ST

03/14/03

U.S. AIR FORCE

Outline

- On Architectures
- The C4ISR Architecture Framework
- C4ISR Architecture Development
- Architecture Evaluation
- Research Issues
- Conclusions

Rationale

Motivation for requiring C4ISR Architecture effort:

- **Need to cope with *uncertainty***
  - **In requirements**
    - **Change in missions**
    - **Wide spectrum of operations (Combined, Joint)**
  - **In rapidly evolving technology, and**
  - **In structural change in DOD and the Air Force (Transformation)**

- **Need for *appropriate interoperability***
  - **within the Air Force**
  - **across DOD**
  - **with Allies and Coalition Partners**
  - **with National and International agencies (e.g., NGOs)**

The Silver Bullet

To manage change, we must look for *Problem Invariants.* Architectures were selected as the right invariant.

- A series of studies recommended that DOD undertake the development of architectures as the basis for acquisition
  - USAF Scientific Advisory Board study of summer ’93
  - Army Science Board study, early ’94
  - Defense Science Board study, 94-95
- The *C4ISR Architecture Framework (version 2.0)* was issued in December 97; ASD(C3I), J6, and USD(AT&L) in 2/98 directed that it be used for all new C4ISR systems
- Memorandum dated 21 March 2000 broadened scope of C4ISR AF to DoD Architecture Framework
- At the DoD CIO's direction (ASD(C3I)), *DoD Architecture Framework 1.0* is being developed; it is scheduled to be completed late in 2002.

DOD Strategic Direction

- **From the ASD(C3I), J6, and USD(AT&L) 2/28/98 memorandum:**

> ***"We see the C4ISR Architecture Framework as a critical element of the strategic direction in the Department, and accordingly direct that all on-going and planned C4ISR or related architectures be developed in accordance with Version 2.0. Existing C4ISR architectures will be redescribed in accordance with the Framework during appropriate revision cycles."***

Issues

- The C4ISR Architecture Framework, v. 2, was an untested draft that was cast (inadvertently) in concrete.
  - The Framework is sound, done by experienced persons
  - It contains several implicit but deliberate features that are not well understood by the community; the result has been the expenditure of resources with very little useful outcome (the efforts eventually deadlock)
  - It focuses on products for representing the architecture, but not on processes for developing architectures
  - It does not recommend a methodology, but implicitly excludes methodologies by the way products are described
  - Planned revisions/updates were not carried through
  - It does not address temporal and evaluation issues

Definitions

- **An architecture is defined as the structure of components, their relationships, and the principles and guidelines governing their design and evolution over time**
  - **IEEE STD 610.12 as stated in the C4ISR Architecture Framework v. 2.0**
- ΑΡΧΙΤΕΚΤΩΝ ***(Greek)*** **= Master Builder**
- **Systems "Architecting" is the process of creating (conceptualizing, designing, and building) UNPRECEDENTED, COMPLEX systems***
- **Use of an architecture is essential when customer/user needs are ill-structured and the likely system unprecedented and complex**
- **Architecting is an Art as well as a Science**

** E. Rechtin in Systems Architecting: Creating and Building Complex Systems*

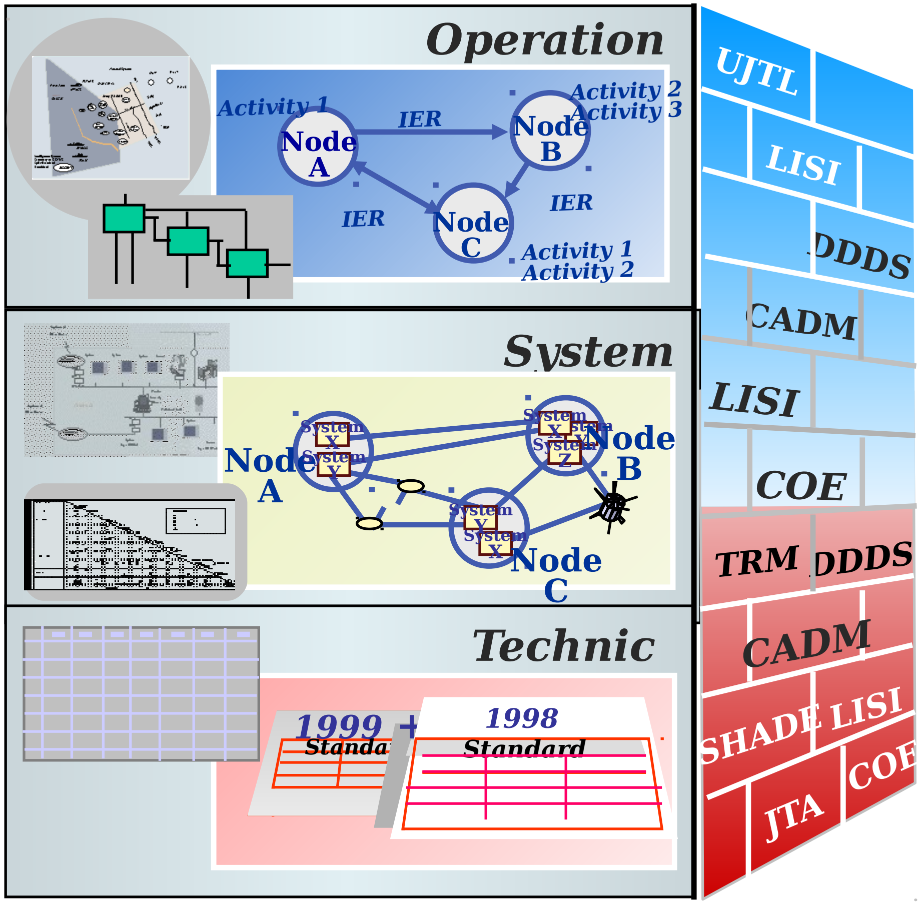
Purpose of a C4ISR Architecture

" The purpose of C4ISR architectures is to improve capabilities by enabling the quick synthesis of "go-to-war" requirements with sound investments leading to the rapid employment of improved operational capabilities, and enabling the efficient engineering of warrior systems."

- C4ISR Architecture Framework, Version 2.0

Components of the Framework

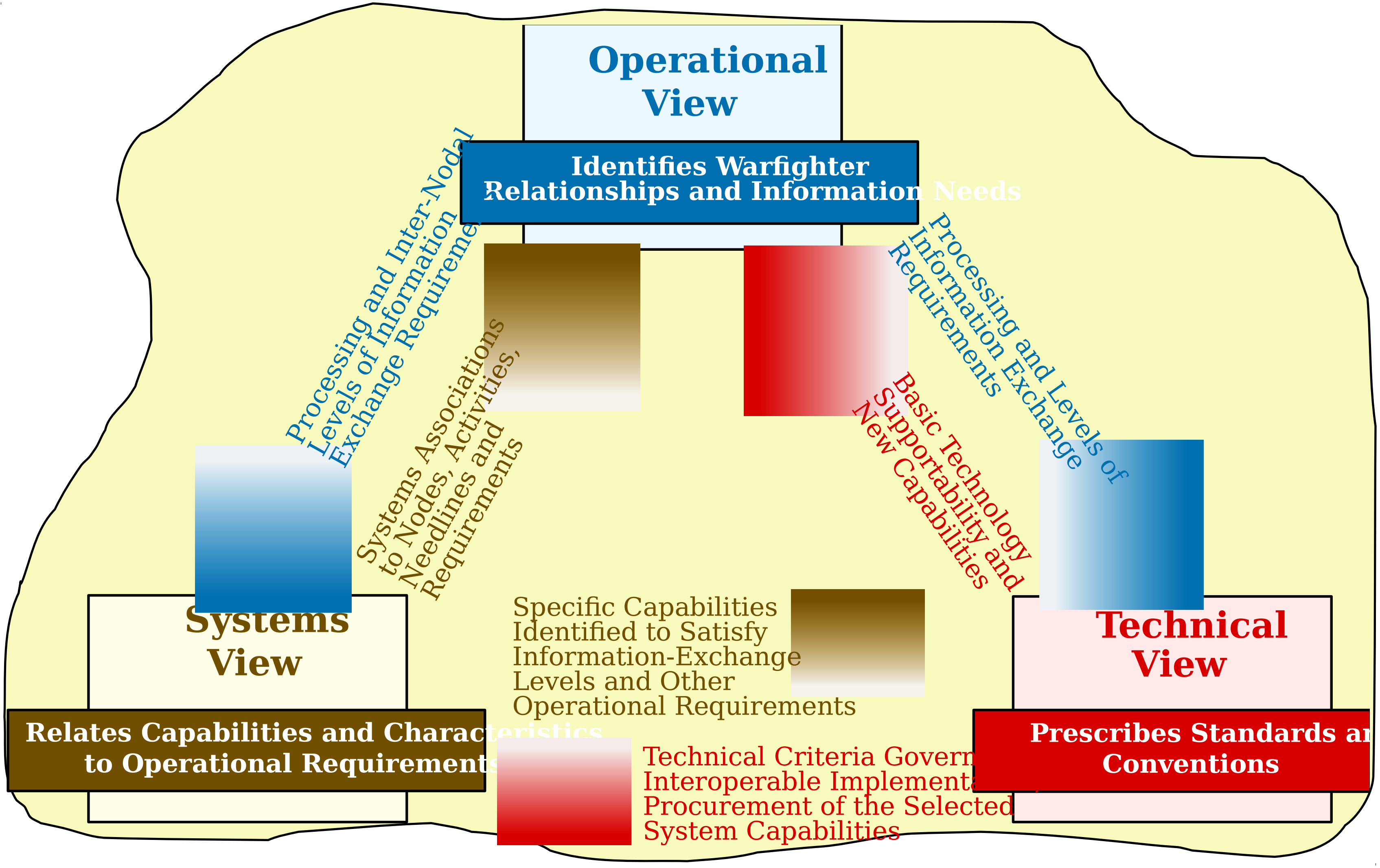
- **Common Definitions**
- **Common Products and Data**
- **Common References**



Architecture Views

- **The *operational architecture view* is a description of the tasks and activities, operational elements, and information flows required to accomplish or support a military operation.**
- **The *systems architecture view* is a description, including graphics, of systems and interconnections providing for, or supporting, warfighting functions.**
- **The *technical architecture view* is the minimal set of rules governing the arrangement, interaction, and interdependence of system parts or elements, whose purpose is to ensure that a conformant system satisfies a specified set of requirements.**

Inter-relationships between views



Operational View

Identifies Warfighter Relationships and Information Needs

Processing and Inter-Nodal Levels of Information Exchange Requireme

Systems Associations to Nodes, Activities, Needlines and Requirements

Processing and Levels of Information Exchange Requirements

Basic Technology Supportability and New Capabilities

Systems View

Relates Capabilities and Characteristics to Operational Requirements

Specific Capabilities Identified to Satisfy Information-Exchange Levels and Other Operational Requirements

Technical Criteria Govern
Interoperable Implementa
Procurement of the Selected
System Capabilities

Technical View

Prescribes Standards an
Conventions

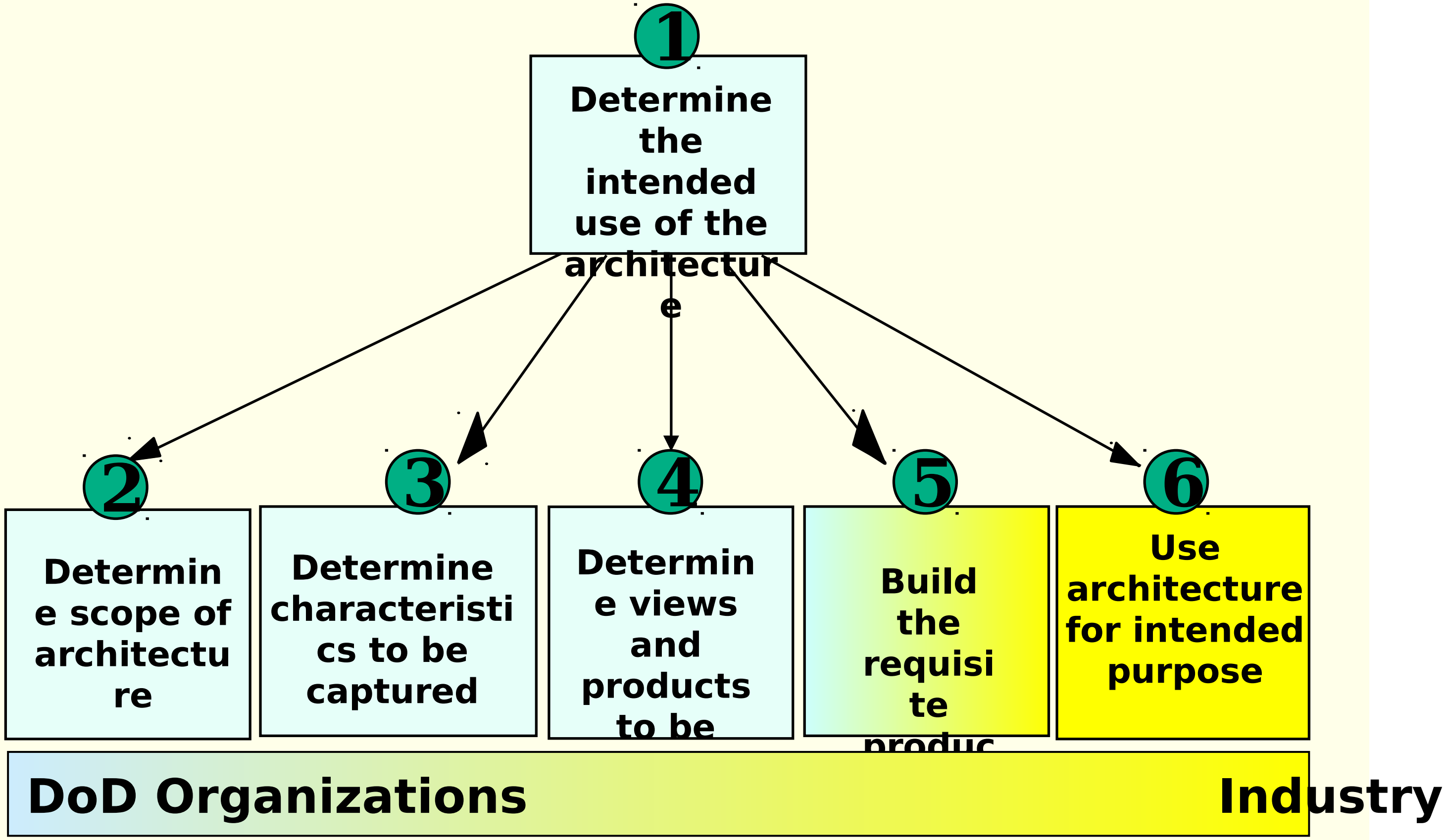
The Problem

- Operational Architecture Views are developed by Operators (MAJCOMS; AC2ISRC)
- System Architecture Views are developed by Product Centers (e.g., ESC)
- Technical Architecture views are done by DISA, AF/SC, AFCA
- These views cannot be developed sequentially; the process deadlocks half-way
- They are NOT three architectures, they are different views of a single architecture
- Consequently, there must be a SINGLE architect who has responsibility for each architecture development and coordinates the work of the Operators (operational view), the Systems people (Systems View) and the IT people (Technical view) and is accountable to the person/ organization that commissioned the architecture

Architecture Development Guidelines

- Architectures should be built with a purpose in mind.
  - If groups of architectures built by various organizations are to be compared, it is important that they all be built from the start with the purpose of comparison in mind.
- Architectures should facilitate, not impede, communication among humans
- Architectures should be relatable, comparable across DoD.

Universal Guidance



1 Determine the intended use of the architecture

2 Determine scope of architecture

3 Determine characteristics to be captured

4 Determine views and products to be

5 Build the requisite produc

6 Use architecture for intended purpose

DoD Organizations

Industry

The Framework provides “instructions” to the architect

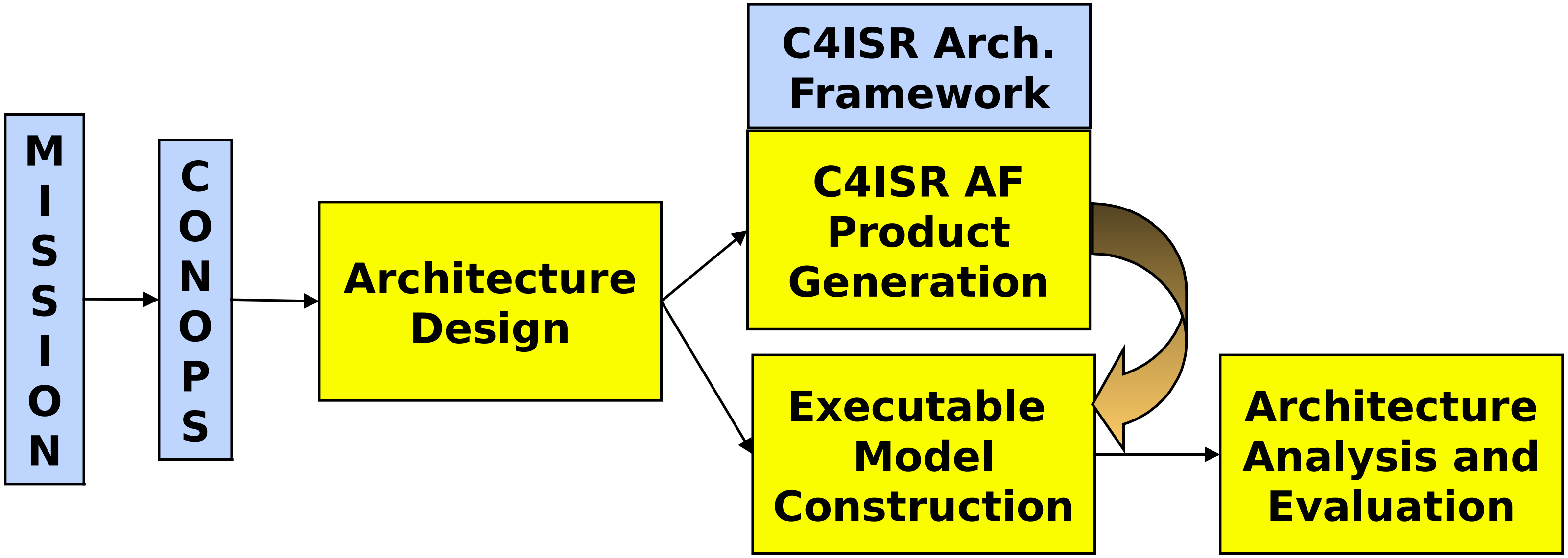
without prescribing specific methodologies or tools

To Develop an Architecture

- ❑ Determine the intended use of the architecture
- ❑ Determine the architecture scope
  - ➢ Establish the Point of View
  - ➢ Establish the Boundary of the architecture
  - ➢ Establish the Layer (Domain, CONOPS,…)
  - ➢ Establish the Time Frame (as is, to be, …)
- ❑ Determine the characteristics to be captured
- ❑ Determine views and products to be built
- ❑ Build the requisite products
- ❑ Carry out Analysis, Evaluation, and Comparison
- ❑ Use architecture for intended purpose
  - ➢ To make acquisition decisions
  - ➢ To design systems
  - ➢ To migrate systems

U.S. AIR FORCE

The Approach


MISSION
CONOPS
Architecture Design
C4ISR Arch. Framework
C4ISR AF Product Generation
Executable Model Construction
Architecture Analysis and Evaluation

- Alternative approaches and tools can be used to design the C4ISR Architecture views – not an issue
- The executable model is used for behavior and performance evaluation
- There are many architectural efforts at different levels and for different problems, but these efforts are usually “architecture product based” and not goal or problem based

Process for C4ISR Arch. Development

- Two six stage processes have been developed for generating the Essential/Mandatory and Supporting products for the Operational and Systems architecture views
- One process has been derived by approaching the problem from the systems engineering point of view and using Structured Analysis; another process has been derived using Object Orientation.
- Each product has been perceived as an Entity containing data; a formal Data Model was derived showing the relationship among the various entities
- The relationships among the entities induced a partial ordering which led to a series of steps for their production
- The processes utilize existing tools and techniques to derive the requisite products and are compatible with the development of an Executable model

Key Entities

- Operational Nodes and Operational Elements
- Activities
- Needlines
- Operational Information Elements

(Operational)

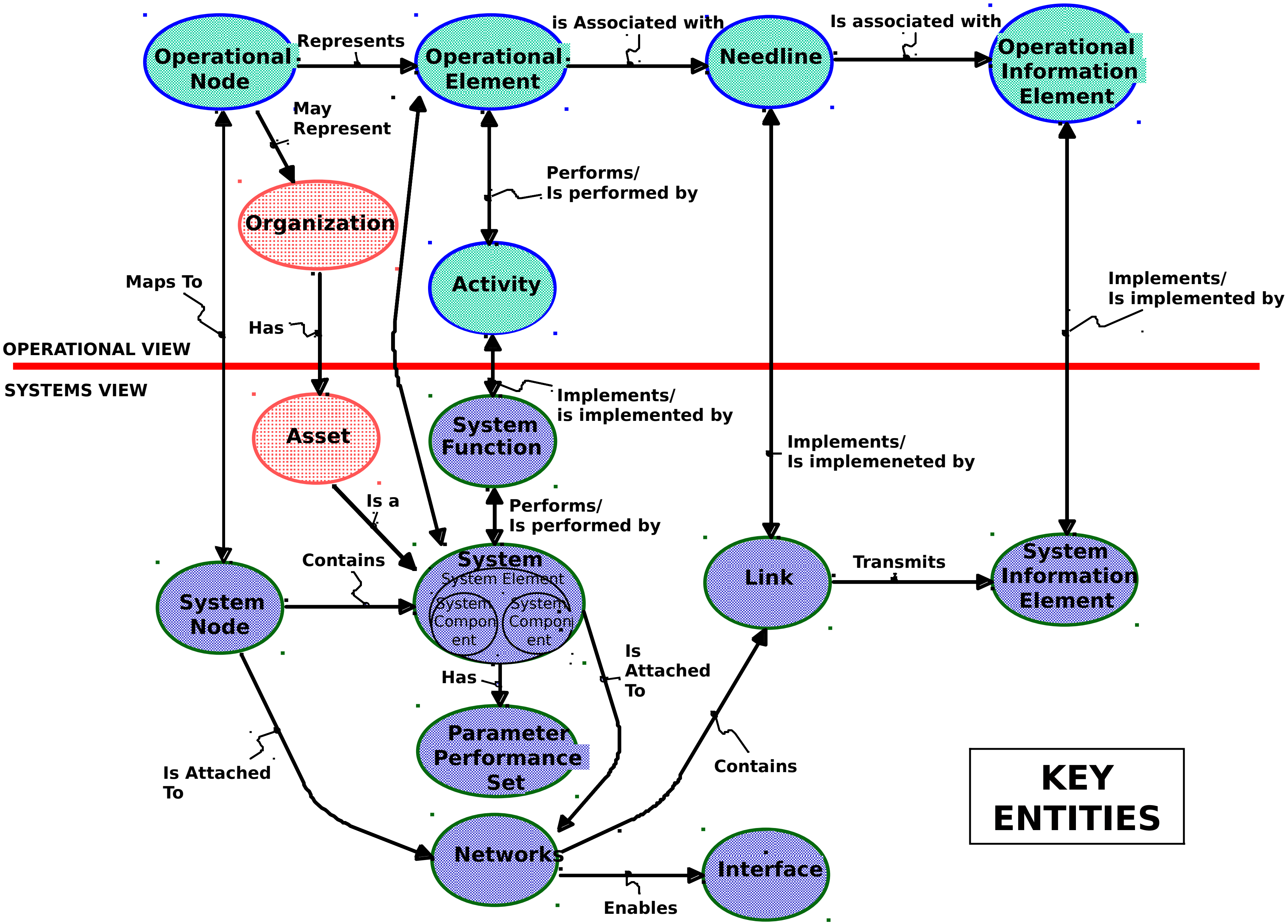
- Organization
- Asset

(Organizational)

- System, System Element, System Component
- System Function
- System Node
- Link
- System Information Element
- Performance Parameter Set
- Networks and Interfaces

(Systems)

KEY ENTITIES



OPERATIONAL VIEW

SYSTEMS VIEW

Operational Node

Represents

Operational Element

is Associated with

Needline

Is associated with

Operational Information Element

May Represent

Organization

Performs/ Is performed by

Activity

Maps To

Has

Implements/ Is implemented by

Implements/ is implemented by

Asset

System Function

Implements/ Is implemeneted by

Is a

Performs/ Is performed by

Contains

System

System Element

System Component

System Component

System Node

Link

Transmits

System Information Element

Has

Is Attached To

Parameter Performance Set

Is Attached To

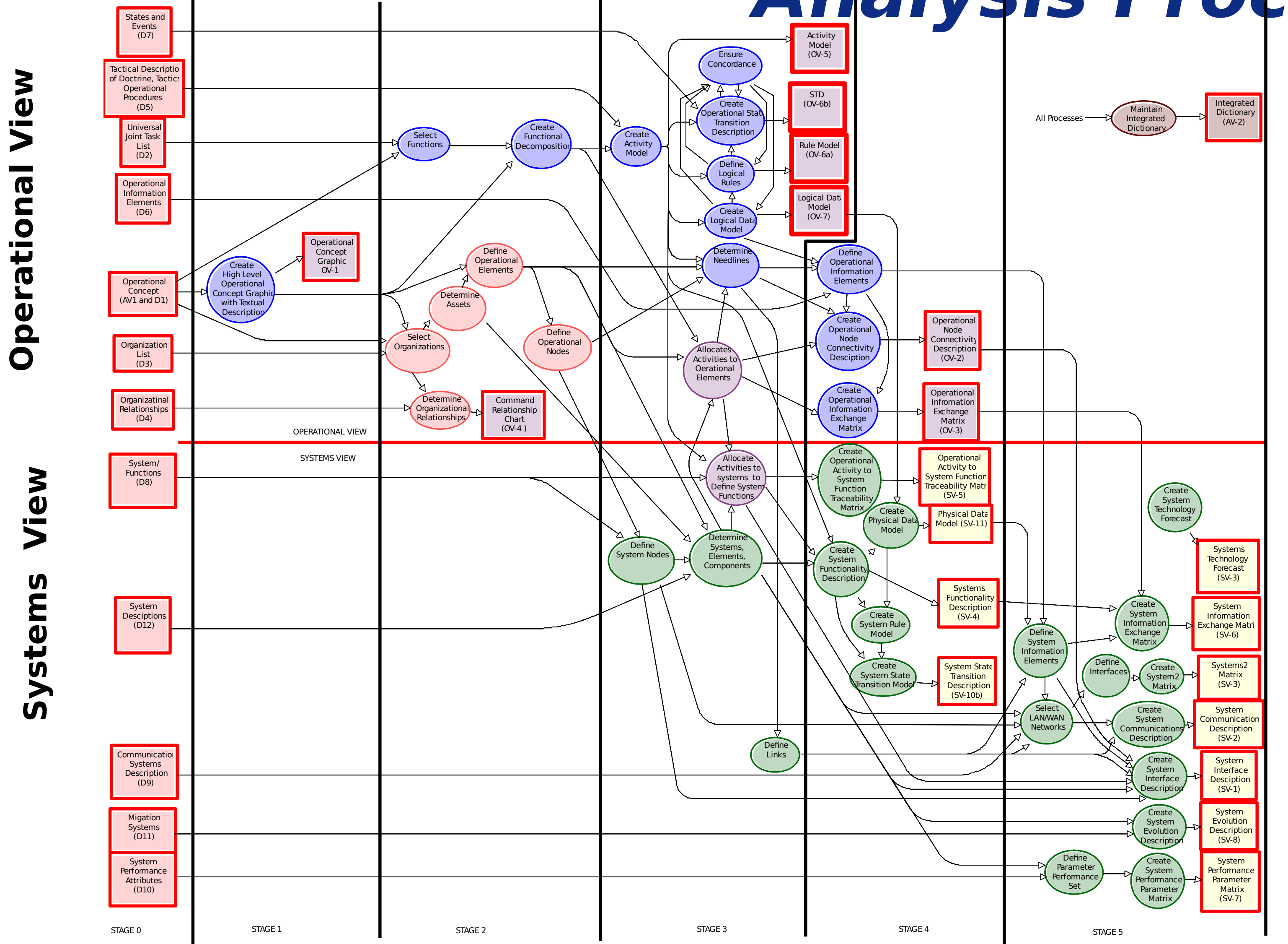
Contains

Networks

Enables

Interface

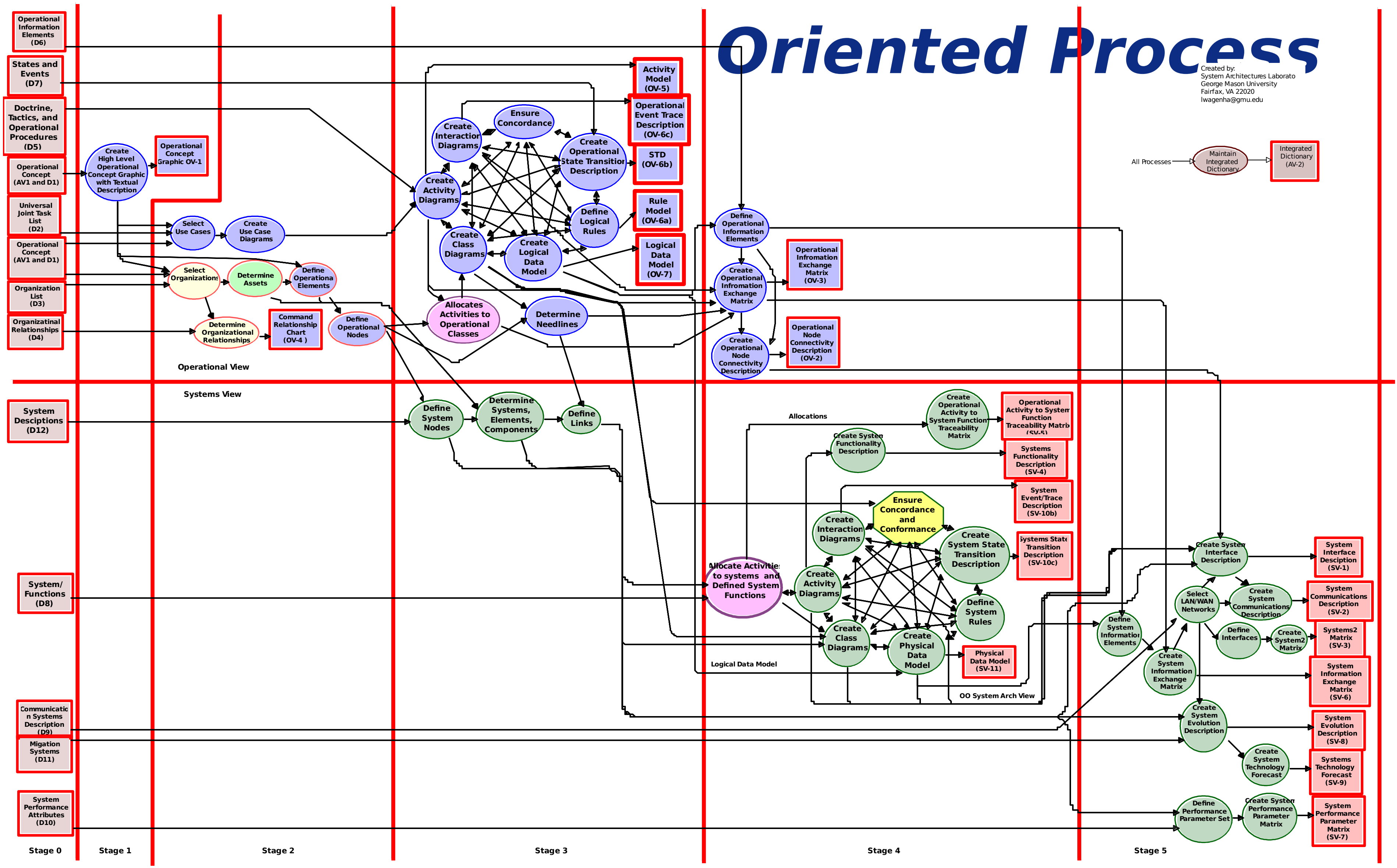
A Six Stage Structured Analysis Process


Operational View
Systems View
States and Events (D7)
Tactical Descriptio of Doctrine, Tactics Operational Procedures (D5)
Universal Joint Task List (D2)
Operational Information Elements (D6)
Operational Concept (AV1 and D1)
Organization List (D3)
Organizatinal Relationships (D4)
System/ Functions (D8)
System Descriptions (D12)
Communication Systems Description (D9)
Migation Systems (D11)
System Performance Attributes (D10)
Create High Level Operational Concept Graphic with Textual Description
Operational Concept Graphic OV-1
Select Functions
Create Functional Decomposition
Define Operational Elements
Determine Assets
Select Organizations
Determine Organizational Relationships
Command Relationship Chart (OV-4)
Define Operational Nodes
OPERATIONAL VIEW
SYSTEMS VIEW
Create Activity Model
Ensure Concordance
Create Operational State Transition Description
Define Logical Rules
Create Logical Data Model
Determine Needlines
Allocates Activities to Oerational Elements
Allocate Activities to systems to Define System Functions
Define System Nodes
Determine Systems, Elements, Components
Define Links
Activity Model (OV-5)
STD (OV-6b)
Rule Model (OV-6a)
Logical Data Model (OV-7)
Define Operational Information Elements
Create Operational Node Connectivity Description
Create Operational Information Exchange Matrix
Create Operational Activity to System Function Traceability Matrix
Create Physical Data Model
Create System Functionality Description
Create System Rule Model
Create System State Transition Model
Operational Node Connectivity Description (OV-2)
Operational Infromation Exchange Matrix (OV-3)
Operational Activity to System Function Traceability Matr (SV-5)
Physical Data Model (SV-11)
Systems Functionality Description (SV-4)
System State Transition Description (SV-10b)
All Processes
Maintain Integrated Dictionary
Integrated Dictionary (AV-2)
Create System Technology Forecast
Systems Technology Forecast (SV-3)
Define System Information Elements
Create System Information Exchange Matrix
System Information Exchange Matri (SV-6)
Define Interfaces
Create Systems2 Matrix
Systems2 Matrix (SV-3)
Select LAN/WAN Networks
Create System Communications Description
System Communication Description (SV-2)
Create System Interface Description
System Interface Description (SV-1)
Create System Evolution Description
System Evolution Description (SV-8)
Define Parameter Performance Set
Create System Performance Parameter Matrix
Systems Performance Parameter Matrix (SV-7)
STAGE 0
STAGE 1
STAGE 2
STAGE 3
STAGE 4
STAGE 5

Legacy system issues force these interactions;
Only elementary schoolbook solutions allow decoupling

A Six Stage Object Oriented Process

Created by:
System Architectures Laborato
George Mason University
Fairfax, VA 22020
lwagenha@gmu.edu



Operational Information Elements (D6)

States and Events (D7)

Doctrine, Tactics, and Operational Procedures (D5)

Operational Concept (AV1 and D1)

Universal Joint Task List (D2)

Operational Concept (AV1 and D1)

Organization List (D3)

Organizatinal Relationships (D4)

System Desciptions (D12)

System/ Functions (D8)

Communicatio n Systems Description (D9)

Migation Systems (D11)

System Performance Attributes (D10)

Create High Level Operational Concept Graphic with Textual Description

Operational Concept Graphic OV-1

Select Use Cases

Create Use Case Diagrams

Select Organizations

Determine Assets

Define Operational Elements

Determine Organizational Relationships

Command Relationship Chart (OV-4)

Define Operational Nodes

Operational View

Systems View

Create Interaction Diagrams

Ensure Concordance

Create Operational State Transition Description

Create Activity Diagrams

Define Logical Rules

Create Class Diagrams

Create Logical Data Model

Allocates Activities to Operational Classes

Determine Needlines

Activity Model (OV-5)

Operational Event Trace Description (OV-6c)

STD (OV-6b)

Rule Model (OV-6a)

Logical Data Model (OV-7)

Define System Nodes

Determine Systems, Elements, Components

Define Links

Define Operational Information Elements

Create Operational Infromation Exchange Matrix

Create Operational Node Connectivity Description

Operational Infromation Exchange Matrix (OV-3)

Operational Node Connectivity Description (OV-2)

Allocations

Create Operational Activity to System Function Traceability Matrix

Operational Activity to System Function Traceability Matrix (SV-5)

Create System Functionality Description

Systems Functionality Description (SV-4)

System Event/Trace Description (SV-10b)

Ensure Concordance and Conformance

Create Interaction Diagrams

Create System State Transition Description

Systems State Transition Description (SV-10c)

Allocate Activities to systems and Defined System Functions

Create Activity Diagrams

Define System Rules

Create Class Diagrams

Create Physical Data Model

Physical Data Model (SV-11)

Logical Data Model

OO System Arch View

All Processes

Maintain Integrated Dictionary

Integrated Dictionary (AV-2)

Create System Interface Description

System Interface Desciption (SV-1)

Select LAN/WAN Networks

Create System Communications Description

System Communications Description (SV-2)

Define System Information Elements

Define Interfaces

Create System2 Matrix

Systems2 Matrix (SV-3)

Create System Information Exchange Matrix

System Information Exchange Matrix (SV-6)

Create System Evolution Description

System Evolution Description (SV-8)

Create System Technology Forecast

Systems Technology Forecast (SV-9)

Define Performance Parameter Set

Create System Performance Parameter Matrix

System Performance Parameter Matrix (SV-7)

Stage 0

Stage 1

Stage 2

Stage 3

Stage 4

Stage 5

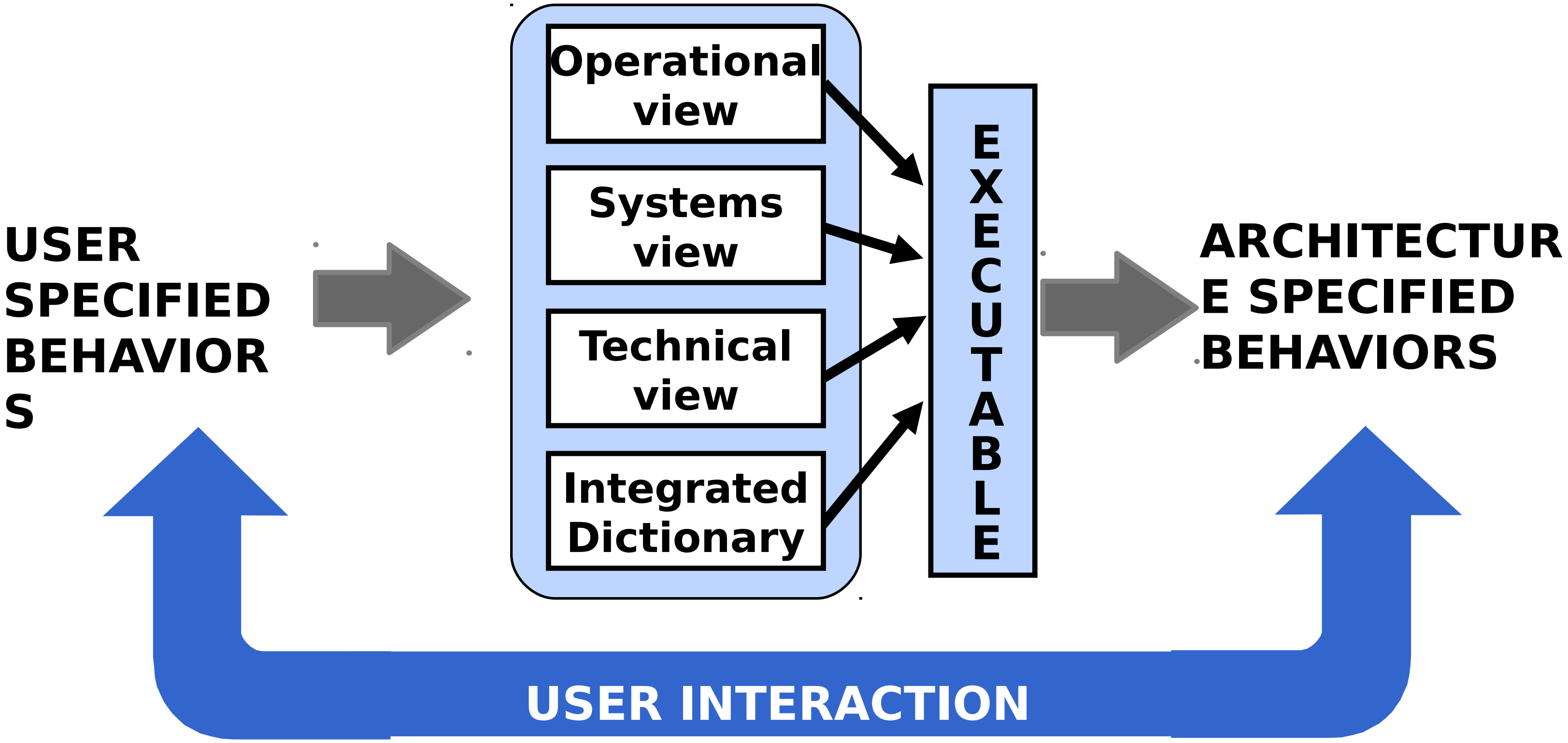
Remarks

- The top half contains activities or transformations related to the Operational Architecture view; the bottom half to the Systems Architecture View
- Note that the two views are crosslinked several times. This was the first lesson learned by those who tried to do C4ISR Architectures: The OA view and the SA View cannot be developed independently from each other
- The crosslinking is in both directions: system information is needed in the OA view (legacy systems; new systems) and activity information in the SA view
- The OA view can be done independently, but only at a high level of abstraction (Domain level). The SA view cannot be done independently of the OA view.
- C4ISR Architecture products are obtained at every step of the process. This means that, if concordance is not maintained rigorously from the start, there will be continuous need to revise and update already developed products

Architecture Evaluation

- To answer the users concerns and show how well the architecture meets their requirements, we need to carry out logical, behavioral and performance analyses to evaluate the architecture
- Three items are needed:
  - A formal set of requirements for the architecture; what must the architecture do, logically and behaviorally, and how well must it do it (performance)
  - A description of the conditions for the life-cycle of the architecture
  - An executable model, based on the information contained in the static views created from the analysis of the mission and operational concept.
- Traceability between the static OA, SV products and the executable model is critical

A Better Perspective



- User behavioral and performance requirements vs. architecture behavior and performance

Research Issues – Near Term

- Executable Models
  - Not the same as simulation models!
  - Executable models in the context of architectures have been defined by BMDO (now MDA) as dynamic models derived formally from the architecture diagrams with complete traceability (audit trail)
  - An executable model is used both for analysis (it is a mathematical model) and for simulation
  - The existing commercial tools do not produce an executable model from the architecture diagrams
  - In object oriented approaches, there are several diagrams that can be used as the basis of the executable – which is better for what?
  - Tools for executable models (CPN, FSA, etc.) do not interoperate with architecture design tools
  - Proof of Concept: In 1999/2000 it was shown that this is possible (Ptech & GMU SBIR Phase I, AFRL/IF)

Research Issues – Near Term

- Problems regarding Executable Models
  - Derive an executable model from an Architecture’s Integrated Dictionary
  - Derive an executable model from the Operational Architecture view
  - Derive an executable model from the Systems Architecture view
  - How are the two executable models related when the systems view includes legacy systems?
  - What are the design constraints for achieving concordance between the two executable models?

Research Issues - Near Term

- **Problems regarding Logical Correctness of Architectures**
  - **The architecture definition includes two Rule Models (operational and systems views). We need to establish that the rules are consistent and coherent.**
    - **Possible Approach based on Petri Net representation of rules (*A. K. Zaidi and A. H. Levis, "Validation and Verification of Decision Making Rules," Automatica, Vol. 33, No. 2, Feb. 1997, pp. 155-169*)**
  - **Is the operational architecture view fully reflected in the systems architecture view?**
    - **Possible approach based on deconstruction of operational arch. Petri Net and systems architecture Petri Net *(F. Valraud and A. H. Levis, "On the Quantitative Evaluation of Functionality in C3 Systems," in <u>Information Technology for Command and Control</u>, S. J. Andriole and S. M. Halpin, Eds., IEEE Press, New York 1991)***

Conclusion

- We know how to design C4ISR architectures using either structured analysis or object orientation and generate the requisite products (for C4ISP, for JROC…)
- There are existing COTS tools that can support architecture development (many tools of different capability and different levels of sophistication – issue: training staff to use them)
- But the architecture development problem has not been formulated and managed properly:
  - No single architect is assigned responsibility for all views (integrated approach – not integrated architectures)
  - The purpose of the architecture is not articulated
  - Operational concepts often are not specified
  - The difference between traditional systems engineering approaches and the C4ISR Architecture Framework has not been considered
- Consequently, we have a myriad of incomplete architecture efforts that are not producing the expected benefits
- It is time to change that!

Headquarters U.S. Air Force

Integrity - Service - Excellence

C4ISR ARCHITECTURES AND A PROCESS FOR THEIR DESIGN

Alexander H. Levis
AF/ST

03/14/03

U.S. AIR FORCE